Directional Sensor Control: Heuristic Approaches

Shankarachary Ragi, *Student Member, IEEE,* Hans D. Mittelmann, and Edwin K. P. Chong, *Fellow, IEEE*

***Abstract*—We study the problem of controlling multiple 2-D directional sensors while maximizing an objective function based on the information gain corresponding to multiple target locations. We assume a joint prior Gaussian distribution for the target locations. A sensor generates a (noisy) measurement of a target only if the target lies within the field-of-view of the sensor, where the statistical properties of the measurement error depend on the location of the target with respect to the sensor and the direction of the sensor. The measurements from the sensors are fused to form global estimates of target locations. This problem is combinatorial in nature—the computation time increases exponentially with the number of sensors. We develop heuristic methods to solve the problem approximately, and provide analytical results on performance guarantees. We then improve the performance of our heuristic approaches by applying an approximate dynamic programming approach called rollout. In addition, we address a variant of the above problem, where the goal is to map the sensors to the targets while maximizing the above-mentioned objective function. This mapping problem also turns out to be combinatorial in nature, so we extend one of the above heuristics to solve this mapping problem approximately. We compare the performance of these heuristic approaches analytically and empirically.**

***Index Terms*—Directional sensor control, maximizing information gain, rollout on heuristics, mapping sensors to targets.**

I. Introduction

Directional sensors constitute a class of sensors that have a limited field-of-view, e.g., surveillance cameras, phased-array antennas, infrared sensors, and ultrasound sensors. The methods to control the directions of these sensors are gaining importance owing to a wide range of applications, including surveillance, detection, and tracking. In this study, we develop tractable solutions to the problem of assigning directions to multiple 2-D directional sensors to maximize the information gain corresponding to multiple target locations. Directional sensor control has been studied before in various contexts [2]–[5]; a general survey of this topic can be found in [6], where the focus is on coverage issues in directional sensor networks.

In this study, we assume a joint prior Gaussian distribution for the target locations, and we assume that the sensor locations are known exactly. A directional sensor has a limited field-of-view (FOV), where the area sensed by the sensor is given by a sector in a circular region around the sensor as depicted in Figure 1. The direction of the FOV of a sensor can be changed by appropriately changing the direction of the sensor. The direction of a sensor can take several discrete values in the interval $[0, 2\pi)$. A directional sensor generates a measurement of a target if and only if the target lies within the FOV of the sensor. We assume that there is a notional fusion center, which collects the measurements from each sensor, and fuses them to form global estimates of target locations.

The objective is to assign a direction to each sensor while maximizing an objective function based on the information gain corresponding to target locations. This problem is hard to solve exactly because of its combinatorial nature—the computation time increases exponentially with the number of sensors. In this study, we develop heuristic approaches that are tractable, and provide bounds on the optimal information gain. We apply rollout on these heuristic approaches (as in [7]) via a dynamic programming formulation [8] to improve the performance of our heuristics with respect to the above objective function.

We then address the above problem using a different formulation, where the objective is to find a mapping of sensors to targets while maximizing the above-mentioned objective function. This problem is also combinatorial in nature, so we extend one of the heuristic approaches, developed for the previous formulation, to solve the mapping problem approximately.

Section II provides a detailed description of the problem. In Section III, we discuss various heuristic approaches to solve the above problem approximately, and we discuss natural sufficient conditions on objective functions that lead to provable guaranteed performance for our heuristics. However, we show via counterexamples that our objective function does not satisfy these sufficient conditions, suggesting that the nature of the problem is highly nontrivial. Sections IV and V provide simulation results and concluding remarks respectively.

Parts of a preliminary version of this paper were published in the proceedings of 2013 SPIE Optics + Photonics Symposium [1]. This paper differs from the conference version in the following ways: 1) we address our directional sensor control problem via a new formulation, where we map sensors to targets while maximizing an objective function, and identify that this mapping problem is combinatorial in nature, 2) we extend a heuristic approach, published in the conference version, to solve our mapping problem approximately, 3) we provide analytical results comparing the performance of our heuristics under certain sufficient conditions on the objective function, and 4) we show via counterexamples that our objective function does not satisfy these sufficient conditions, which proves that our objective function possesses highly nontrivial features that elude currently known analytical machinery.

An earlier version of this paper was presented at the 2013 SPIE Optics + Photonics Symposium [1] and was published in its Proceedings (DOI: http://dx.doi.org/10.1117/12.2022451).

The work of the first and third authors was supported in part by Colorado State University's Information Science & Technology Center. The work of the second author was supported in part by AFOSR under grant FA9550-12-1-0153.

S. Ragi and E. K. P. Chong are with the Department of Electrical and Computer Engineering, Colorado State University, Fort Collins, CO 80523-1373, USA e-mail: {shankar.ragi, edwin.chong}@colostate.edu.

H. D. Mittelmann is with the School of Mathematical and Statistical Sciences, Arizona State University, Tempe, AZ 85287-1804, USA e-mail: mittelmann@asu.edu.

II. Problem Specification

Targets. There are N targets in 2-D, where $(\chi_1, \chi_2, \ldots, \chi_N)$ represent the locations of the targets. The target locations are not known exactly. However, we assume a joint prior Gaussian distribution for the target locations.

Directional Sensors. There are M directional sensors in 2-D, where $(s_1, s_2, \ldots, s_M)$ represent the locations of the sensors. The sensor locations are known exactly. Let $\Theta = \{1, \ldots, K\}$ be the set of directions each sensor can take, where each direction is a value in the interval $[0, 2\pi)$. Let $u = (u_1, \ldots, u_M)$ be the control vector, where $u_i \in \Theta$, $i = 1, \ldots, M$, is the direction of the ith sensor. The field-of-view of a sensor, pointed at a particular direction $\theta \in \Theta$, is shown in Figure 1, where r and α are the radial and angular sensing ranges of the sensor respectively. Because our focus is on solution methods for the problem of controlling directional sensors, rather than on detailed sensor modeling, we adopt a simple 2-D sensing model as shown in Figure 1. The 2-D conical field-of-view model in Figure 1 is an appropriate approximation to the sensing behavior of many directional sensors, including surveillance cameras [9] and phased arrays [10]. In the case of surveillance cameras, the limited radial range in our sensing model is well justified by the constraint that the camera cannot detect the presence of targets up to a given maximum size located outside a certain range from the camera, when the size of the target image is smaller than a single pixel.

Measurement Errors. Each sensor generates a 2-D position measurement of a target only if the target lies within the FOV of the sensor. These measurements are corrupted by random errors that depend on the relative location of the target with respect to the sensor and the direction of the sensor. The measurement of the jth target at the ith sensor is given by

$$z_{ij} = \begin{cases} \mathbf{H}\chi_j + n_{ij} & \text{if target lies within} \\ & \text{FOV of sensor,} \\ \text{no measurement} & \text{otherwise,} \end{cases}$$

where $n_{ij} \sim \mathcal{N}(0, \mathbf{Z}(s_i, u_i, \chi_j))$, $\mathbf{H}$ is an observation model, and $\mathbf{Z}(\cdot)$ is the measurement error-covariance matrix, which depends on the direction of the sensor and the locations of the target and the sensor.

Fusion. The observations obtained from the sensors are fused to form a global estimate for each target. Let $\mathcal{N}(\xi_j^{\text{prior}}, \mathbf{P}_j^{\text{prior}})$, $j = 1, \ldots, N$, be the prior distributions (Gaussian) of the target-locations. Given the observations and the prior distributions, we evaluate the posterior distribution of the target-locations by fusing the observations. The target observations are not Gaussian; the evaluation of the true Bayesian posterior distribution is not tractable. Therefore, we approximate the posterior distribution of jth target as $\mathcal{N}(\xi_j, \mathbf{P}_j)$, $j = 1, \ldots, N$, where ξ_j and $\mathbf{P}_j$ are evaluated according to Algorithm 1, where z_{ij} is the observation generated at sensor i.

Objective. The objective is to compute u, i.e., the directions for the sensors, such that the following objective function

Algorithm 1 Approximate Posterior Distribution

$A = \left[\mathbf{P}_j^{\text{prior}}\right]^{-1}$
$y = A\xi_j^{\text{prior}}$
for $i = 0$ to M **do** ▷ Information filtering equations
  if Sensor i generates observation **then**
    $A \leftarrow A + \mathbf{H}^{\mathrm{T}} \left[\mathbf{Z}\left(s_i, u_i, \xi_j^{\text{prior}}\right)\right]^{-1} \mathbf{H}$
    $y \leftarrow y + \mathbf{H}^{\mathrm{T}} \left[\mathbf{Z}\left(s_i, u_i, \xi_j^{\text{prior}}\right)\right]^{-1} z_{ij}$
  end if
end for
$\mathbf{P}_j = A^{-1}$
$\xi_j = A^{-1}y$

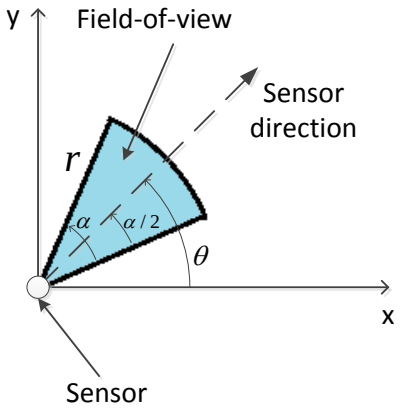

Fig. 1. Field-of-view of a sensor

(based on information gain) is maximized:

$$-\mathrm{E}\left[\sum_{j=1}^{N} \log\det(\mathbf{P}_j)\right],$$

where the expectation is over the prior joint-distribution of target locations, and $\mathbf{P}_j$ is the posterior distribution of the jth target, which is evaluated using Algorithm 1 given the locations of the targets—these target locations are used only to check if the targets fall within the FOV of the sensors.

Information theory provides a way of quantifying the amount of signal-related information that can be extracted from a measurement. This theory also provides tools for assessing the fundamental limitations of different measurement systems in achieving objectives such as detection and tracking, and these fundamental limits can be related to the amount of information gain associated with a specific measurement method. This motivated us to choose an information-theoretic objective function (see [11] for more arguments on behalf of using an information-theoretic objective function).

Optimal Solution. The optimal directions for the sensors are given by

$$u^* = \arg\max_{u \in \Theta^M} R(u), \tag{1}$$

where

$$R(u) = -\mathrm{E}\left[\sum_{j=1}^{N} \log\det(\mathbf{P}_j(u))\right] \tag{2}$$

and $\mathbf{P}_j(u)$, $j = 1, \ldots, N$, are evaluated according to Algorithm 1 given the control vector u. We approximate the expectation by a Monte Carlo method. Specifically, we generate several samples from the joint prior distribution of the target locations, and we compute the average (over the samples) objective function value for a given control action. The above problem is a combinatorial optimization problem, where the computational time required to find the optimal solution is $O(K^M)$. Since the computational time increases exponentially with the number of sensors M, we are interested in deriving tractable heuristic methods that are polynomial with respect to the number of sensors.

III. Approximate Solutions

A. Continuous Optimization

We obtain an upper bound on the optimal objective function value by "relaxing" the discrete property of our problem and solving its continuous version. The continuous version of our combinatorial optimization problem is stated as follows:

$$\begin{aligned} &\underset{(u_1,\ldots,u_M)}{\text{maximize}} \quad R(u_1, \ldots, u_M) \\ &\text{subject to} \quad 0 \le u_i < 2\pi,\ i = 1, \ldots, M. \end{aligned}$$

The optimal objective function value stands as an upper bound to the optimal objective function value of our original problem. The solution to the above problem can be obtained via a nonlinear programming (NLP) solver. Specifically, we adopt the simulated annealing algorithm to solve the above problem.

In the following subsection, we present several heuristic approaches to solve our problem approximately. The solutions of these heuristics provide a lower bound on the optimal objective function value (discussed later). Because it is hard to compute the optimal objective function value, we use the upper bound above on the optimal objective function value, i.e., the solution from the above-mentioned continuous optimization approach, to see how close the heuristics are to the optimal.

B. Heuristic Approaches

Let $u = (u_1, u_2, \ldots, u_M)$ represent a solution to our problem. The optimal solution is given by

$$u^* = \arg \max_{u_i \in \Theta, i=1\ldots,M} R(u_1, \ldots, u_M),$$

where $\Theta = \{1, \ldots, K\}$. As we discussed earlier, the computation complexity to obtain the optimal solution through (1) is $O(K^M)$, which is exponential in the number of sensors M. So, we are interested in developing heuristic approaches that are polynomial in the number of sensors. The following algorithm, called $\mathcal{H}_1$, generates the solution $u_{\mathcal{H}_1} = (\bar{u}_1, \ldots, \bar{u}_M)$, where

$$\begin{aligned} \bar{u}_1 &= \arg\max_{u\in\Theta} R(u, \emptyset, \ldots, \emptyset), \\ &\vdots \\ \bar{u}_k &= \arg\max_{u\in\Theta} R(\bar{u}_1, \ldots, \bar{u}_{k-1}, u, \emptyset, \ldots, \emptyset), \\ &\vdots \\ \bar{u}_M &= \arg\max_{u\in\Theta} R(\bar{u}_1, \ldots, \bar{u}_{M-1}, u), \end{aligned}$$

where $\emptyset$ at any jth location in the solution $u = (u_1, \ldots, u_M)$, i.e., u_j replaced by $\emptyset$, means that the senor j is ignored while computing the objective function (as if the sensor j does not generate any observations and does not influence the objective function). Therefore, the algorithm $\mathcal{H}_1$ generates an approximate solution $u_{\mathcal{H}_1} = (\bar{u}_1, \ldots, \bar{u}_M)$, and the objective function value from $\mathcal{H}_1$ is $R(\bar{u}_1, \bar{u}_2, \ldots, \bar{u}_M)$. The computational complexity of $\mathcal{H}_1$ is $O(KM)$, which is now linear with respect to the number of sensors.

We now present a second heuristic approach $\mathcal{H}_2$, where $\mathcal{H}_2$ generates the solution $u_{\mathcal{H}_2} = (\hat{u}_1, \ldots, \hat{u}_M)$. In contrast to the previous heuristic approach, this approach may not generate the elements of the solution vector in the order $\hat{u}_1, \ldots, \hat{u}_M$. Let $(p_1, \ldots, p_M)$ be the order in which the elements of the solution vector are generated, where $(p_1, \ldots, p_M)$ is a permutation of $(1, \ldots, M)$. This algorithm is evaluated stepwise, with a total of M steps. At each step, a sensor is assigned a direction, and this assignment is fixed for the rest of the steps, as described below.

1) In the first step, for each sensor i, we associate a direction that maximizes the objective function that depends only on the direction of sensor i, i.e., we ignore the rest of the sensors when we are associating a direction to the sensor i. Let p_1 and $\hat{u}_{p_1}$ be the sensor and its associated direction respectively that gives the maximum objective function value among all the associations. Now, we assign direction $\hat{u}_{p_1}$ to sensor p_1, and this assignment is fixed for the rest of the steps in the algorithm.
2) In the kth step, we associate a direction to each sensor $j \in \{1, \ldots, M\} \setminus \{p_1, \ldots, p_{k-1}\}$, while maximizing the objective function, which now depends on the directions of the sensors $p_1, \ldots, p_{k-1}$ (computed in the previous $k-1$ steps) and the direction of sensor j. Let $(p_k, \hat{u}_{p_k})$ be the sensor-direction pair that has the maximum objective function value from the above computation. At kth step, we assign direction $\hat{u}_{p_k}$ to the sensor p_k.
3) We repeat the above step until the last sensor p_M is assigned a direction.
4) At the end of the algorithm, we are left with the solution $(\hat{u}_1, \ldots, \hat{u}_M)$.

The computational complexity of this approach is $O(KM^2)$. In both the heuristics $\mathcal{H}_1$ and $\mathcal{H}_2$, since we do not search all possible directions exhaustively, the objective function values from these approaches stand as lower bounds on the optimal value.

Note that the algorithm $\mathcal{H}_1$ requires an ordering among the

sensors, and the objective function value from $\mathcal{H}_1$ depends on this order. But the objective function value from $\mathcal{H}_2$ is independent of the ordering of sensors as $\mathcal{H}_2$ does not require any ordering among the sensors.

C. Rollout on a Heuristic Approach

Given a heuristic approach that solves a combinatorial optimization problem step-wise like our $\mathcal{H}_1$, the authors of [7] have utilized dynamic programming formulation [8] to improve the performance of the heuristic. Specifically, they use an approximate dynamic programming approach called rollout to improve the performance of the given heuristic. We adopt a similar technique, and apply rollout on our heuristics to improve their performance (with respect to the objective function value).

We can obtain the exact optimal solution (step-wise) using the dynamic programming [8] approach as follows. We start at a dummy (artificial) initial state; the state of the algorithm at the 1st stage is (u_1). The state (of the algorithm) at the kth stage is of the form $(u_1, \ldots, u_k)$, also called k-solution. The terminal state is $(u_1, \ldots, u_M)$. The control variable at state $(u_1, \ldots, u_{k-1})$ is $u_k \in \Theta$. We get a reward at the end of the Mth step called terminal reward, which is given by our original objective function $R(u_1, \ldots, u_M)$. Let $J^*(u_1, \ldots, u_k)$ be the optimal value-to-go (see [7] for details) starting from the k-solution, which is the optimal terminal reward given that $(u_1, \ldots, u_k)$ are already assigned to the sensors $1, \ldots, k$. The optimal solution to our problem $(u_1^*, u_2^*, \ldots, u_M^*)$ can be obtained from the following equations:

$$u_k^* = \arg\max_{u \in \Theta} J^*(u_1^*, \ldots, u_{k-1}^*, u), \quad k = 1, \ldots, M. \quad (3)$$

In general, the optimal value-to-go $J^*(\cdot)$ is hard to obtain, which is in fact true for our problem. For practical purposes, $J^*(\cdot)$ is replaced with a heuristic value-to-go $\bar{J}(\cdot)$, which is usually easy to obtain.

Let $\mathcal{H}$ be any heuristic algorithm, which generates the path of states $(\bar{i}_1, \bar{i}_2, \ldots, \bar{i}_M)$, where $\bar{i}_k = (\bar{u}_1, \ldots, \bar{u}_k)$. Let $\bar{J}(\bar{i}_k)$ represent the heuristic value-to-go starting from the k-solution $\bar{i}_k = (\bar{u}_1, \ldots, \bar{u}_k)$, from the algorithm $\mathcal{H}$, i.e., we use $\mathcal{H}$ to evaluate the value-to-go. The value-to-go from the algorithm $\mathcal{H}$ is equal to the terminal reward obtained from the algorithm $\mathcal{H}$, i.e., $\bar{J}(\bar{i}_k) = R(\bar{u}_1, \bar{u}_2, \ldots, \bar{u}_M)$. Therefore, the following is true: $\bar{J}(\bar{i}_1) = \bar{J}(\bar{i}_2) = \ldots = \bar{J}(\bar{i}_M)$. We use this heuristic value-to-go in (3) to find an approximate solution to our problem. We call this approximation algorithm "Rollout on $\mathcal{H}$" ($\mathcal{RH}$ in short; the same notation was used in [7]) due to its structure, which is similar to an approximate dynamic programming approach called *rollout*. The $\mathcal{RH}$ algorithm starts with the original dummy state, and generates the path $(i_1, i_2, \ldots, i_M)$ according to the following equation:

$$i_k = \arg\max_{j \in N(i_{k-1})} \bar{J}(j), \quad k = 1, \ldots, M$$

where, $i_{k-1} = (u_1, \ldots, u_{k-1})$, and

$$N(i_{k-1}) = \{(u_1, \ldots, u_{k-1}, u) | u \in \Theta\}, \quad k = 1, \ldots, M.$$

The following lemma is adapted from [7]. For completeness, we provide its proof.

Lemma 3.1: The algorithm $\mathcal{RH}$ is sequentially improving with respect to $\mathcal{H}$, i.e., $\bar{J}(i_1) \leq \bar{J}(i_2) \leq \ldots \leq \bar{J}(i_M)$, where $(i_1, i_2, \ldots, i_M)$ is the path generated by the $\mathcal{RH}$ algorithm.

Proof: Let $(i_1, i_2^{i_1}, \ldots, i_M^{i_1})$ be the complete solution from $\mathcal{H}$ given i_1, which is obtained from the first step of $\mathcal{RH}$. We can easily verify that $\bar{J}(i_1) = \bar{J}(i_2^{i_1})$. Let i_2 be the solution obtained from the second step of $\mathcal{RH}$, i.e.,

$$\bar{J}(i_2) = \max_{j \in N(i_1)} \bar{J}(j).$$

But

$$\max_{j \in N(i_1)} \bar{J}(j) \geq \bar{J}(i_2^{i_1}) = \bar{J}(i_1).$$

Therefore, $\bar{J}(i_2) \geq \bar{J}(i_1)$. We can extend this argument for the rest of the steps in $\mathcal{RH}$, which proves the result $\bar{J}(i_1) \leq \bar{J}(i_2) \leq \ldots \leq \bar{J}(i_M)$. ■

The authors of [7] have argued that rollout on a heuristic performs no worse than the heuristic in a different context. We will now extend this argument to our problem and prove that $\mathcal{RH}$ outperforms $\mathcal{H}$.

Theorem 3.2: The algorithm $\mathcal{RH}$ outperforms $\mathcal{H}$, i.e., $R(\bar{i}_M) \leq R(i_M)$, where $\bar{i}_M$ and i_M are the final paths generated by the algorithms $\mathcal{H}$ and $\mathcal{RH}$ respectively.

Proof: We can easily verify that $\bar{J}(\bar{i}_1) = \bar{J}(\bar{i}_2) = \ldots = \bar{J}(\bar{i}_M) = R(\bar{i}_M)$. Since $(i_1, \ldots, i_M)$ is the path generated by $\mathcal{RH}$, and since

$$i_1 = \arg\max_{j \in \Theta} \bar{J}(j),$$

the following is true: $\bar{J}(i_1) \geq \bar{J}(j)$ for all $j \in \Theta$. Since $\bar{i}_1 \in \Theta$, therefore $\bar{J}(i_1) \geq \bar{J}(\bar{i}_1)$. Therefore, from the above result and Lemma 3.1, we can obtain the following result:

$$R(i_M) = \bar{J}(i_M) \geq \ldots \geq \bar{J}(i_1) \geq \bar{J}(\bar{i}_1) = R(\bar{i}_M).$$

■

The above result proves that applying rollout on a heuristic approach guarantees to improve the performance of the heuristic with respect to any objective function. The above rollout approach can be viewed as a one-step lookahead approach (or simply one-step rollout), as we optimize, at every stage, the control for the current step by maximizing the value-to-go given the control for the current step. At the expense of increased computational burden, we can further improve the solution from the above rollout by the following approach: optimize the controls for the current and the next steps combined (i.e., for two steps) by maximizing the value-to-go given the controls for the current and the next steps. This can be viewed as a two-step rollout. Similarly, we can generalize this to an m-step rollout; however as m increases, the computational requirement also increases. When $m = M$, the rollout approach finds the exact optimal solution by exhaustively searching through all possible directions, with computational complexity $O(K^M)$ as in the case of (1).

D. Mapping of Sensors to Targets

In this subsection, our problem formulation differs from the formulation in the previous section. Here, our goal is to map the sensors to the targets while maximizing the objective function defined in the previous subsection. Mapping a sensor

to a target gives rise to an assignment of a specific direction to the sensor via the following procedure: a direction that minimizes the angular difference between the direction of the sensor and the direction of the mean of the target's a priori distribution with respect to the sensor's location. Essentially, we are again evaluating the directions for sensors, as in the previous subsections, although indirectly via evaluating a mapping from the set of sensors to the set of targets. The motivation behind formulating this mapping problem is that if the number of targets is less than the number of directions a sensor can take, then the set of feasible solutions is smaller for this new formulation of the problem.

The above mapping problem is also a combinatorial optimization problem, where the computational complexity is now $O(N^M)$, where N is the number of targets and M is the number of sensors. Therefore, we can simply use the heuristic approaches developed in the previous section to solve the current problem, where the set of controls for each sensor now is the set of targets, in contrast to sensor directions being controls in the previous problem. For a given mapping from sensors to targets, we compute the objective function value as described in (2) given the directions of sensors, which are indirectly obtained from a given mapping of sensors to targets.

Let us extend the heuristic algorithm $\mathcal{H}_1$ from the previous section to solve the above problem, and let this new heuristic algorithm be called $\mathcal{MH}_1$ (short for mapping heuristic). In these heuristics, as discussed before, we evaluate directions to sensors stage-wise, i.e., assign a direction to the first sensor, then to the second sensor, and so on. Let u_k be the direction assigned to the kth sensor at stage k. Let

$$U = \{(u_1, \ldots, u_k) | k = 1, \ldots, M, u_k \in \Theta\}$$

be the set of all possible stage-wise controls, where $(u_1, \ldots, u_k)$ at kth stage are the assigned directions to sensors $1, \ldots, k$ respectively. We can notice that the objective function in heuristic algorithm $\mathcal{H}_1$ is defined on the set U.

At this point, it would be interesting to ask if we can provide performance guarantees for $\mathcal{H}_1$ over $\mathcal{MH}_1$. Naturally, such a result would necessitate imposing appropriate restrictions on the objective function. In the following, we will explore what seems to be a reasonable such restriction, based on which we will prove that under such a restriction $\mathcal{H}_1$ outperforms $\mathcal{MH}_1$. However, as we will see later, our problem is sufficiently nontrivial as to frustrate even a reasonable sufficient condition.

To proceed with this investigation, we now provide a definition of a reasonable sufficient condition under which a provable performance guarantee can be achieved.

Definition 1: Given any function $R : U \to \mathbb{R}$, and for every pair of elements in U of the form $(\hat{u}_1, \ldots, \hat{u}_k)$ and $(\bar{u}_1 \ldots, \bar{u}_k)$ for any k that satisfies the condition $R((\hat{u}_1, \ldots, \hat{u}_k)) \geq R((\bar{u}_1 \ldots, \bar{u}_k))$, then R is said to be *continuous monotone* if

$$R((\hat{u}_1, \ldots, \hat{u}_k, a)) \geq R((\bar{u}_1 \ldots, \bar{u}_k, a)),$$

for every $a \in \Theta$.

Example Let $r_i : \Theta \to \mathbb{R}$ for $i = 1, \ldots, M$, and let

$$R((u_1, \ldots, u_k)) = \sum_{i=1}^{k} r_i(u_i),$$

for $k = 1, \ldots, M$. We can easily verify that the above *additive* objective function is continuous monotone.

Let $R : U \to \mathbb{R}$ be any generic objective function, and let $u_{\mathcal{H}_1}$ be the directions for sensors obtained from $\mathcal{H}_1$ and let $u_{\mathcal{MH}_1}$ be the directions for sensors from $\mathcal{MH}_1$ using the objective function R.

Theorem 3.3: If R is continuous monotone, the algorithm $\mathcal{H}_1$ outperforms $\mathcal{MH}_1$ with respect to R, i.e., $R(u_{\mathcal{H}_1}) \geq R(u_{\mathcal{MH}_1})$.

Proof: Let $u_{\mathcal{H}_1} = (\hat{u}_1, \ldots, \hat{u}_M)$ and $u_{\mathcal{MH}_1} = (\bar{u}_1, \ldots, \bar{u}_M)$. The first element of $u_{\mathcal{H}_1}$, i.e., $\hat{u}_1$ is evaluated as follows:

$$\hat{u}_1 = \arg\max_{a \in \Theta} R((a)),$$

whereas $\bar{u}_1$ is obtained (albeit indirectly) by assigning sensor 1 to each target and checking which assignment maximizes the objective function value. In other words, there exists $\Theta_1 \subseteq \Theta$ such that

$$\bar{u}_1 = \arg\max_{a \in \Theta_1} R((a)),$$

which implies that $R((\hat{u}_1)) \geq R((\bar{u}_1))$. Since R is continuous monotone, by definition $R((\hat{u}_1, a)) \geq R((\bar{u}_1, a))$ for every $a \in \Theta$. Therefore, $\max_{a \in \Theta} R((\hat{u}_1, a)) \geq \max_{a \in \Theta} R((\bar{u}_1, a))$. We can extend the above discussion on the first step of $\mathcal{MH}_1$ algorithm to its second step, i.e., there exists $\Theta_2 \subseteq \Theta$ such that $\bar{u}_2 = \arg\max_{a \in \Theta_2} R((\bar{u}_1, a))$. We can easily verify that $\max_{a \in \Theta} R((\hat{u}_1, a)) \geq \max_{a \in \Theta_2} R((\bar{u}_1, a))$ or $R((\hat{u}_1, \hat{u}_2)) \geq R((\bar{u}_1, \bar{u}_2))$. We can extend this argument for the rest of the steps, which proves the result $R((\hat{u}_1, \ldots, \hat{u}_M)) \geq R((\bar{u}_1, \ldots, \bar{u}_M))$. ■

Unfortunately, as alluded to before, things do not turn out as conveniently as one would have desired. More specifically, our objective function (2) is in fact *not* continuous monotone. We will demonstrate this claim with the following counterexample. Nonetheless, the above theorem provides a useful result comparing the performance of $\mathcal{H}_1$ and $\mathcal{MH}_1$ for any generic continuous monotone objective function.

Example Let us consider a scenario with two sensors $\mathbf{a}, \mathbf{b}$ and two targets as shown in Figures 2 and 3. In these figures, the FOV of each sensor is $\pi/5$, i.e., $\alpha = \pi/5$ in Figure 1. We approximate the expectation in the objective function by the following Monte Carlo method. We generate 50 samples from the (joint) target location distribution. For a given control vector u, we compute the objective function $R(u)$ from each sample; the objective function value for the given control vector is given by the average of these 50 objective function values. In these figures, the sensors are represented by small circles, the target (prior) distributions are represented by the error concentration ellipses, and the FOVs are represented by 2-D cones. For this scenario, we compute our objective function values for the following two cases:

1) sensors $\mathbf{a}$ and $\mathbf{b}$ are assigned directions 0 and $\pi/2$ respectively as shown in Figure 2, and

2) sensors $\mathbf{a}$ and $\mathbf{b}$ are assigned directions $-\pi/2$ and $\pi/2$ respectively as shown in Figure 3.

The following are the objective function values for the above cases: $R((0)) = 8.05$, $R((-\pi/2)) = 7.36$, $R((0,\pi/2)) = 9.65$, $R((-\pi/2,\pi/2)) = 10.37$. Therefore, $R((0)) > R((-\pi/2))$, but $R((0,\pi/2)) < R((-\pi/2,\pi/2))$, which proves that our objective function (2) is not continuous monotone.

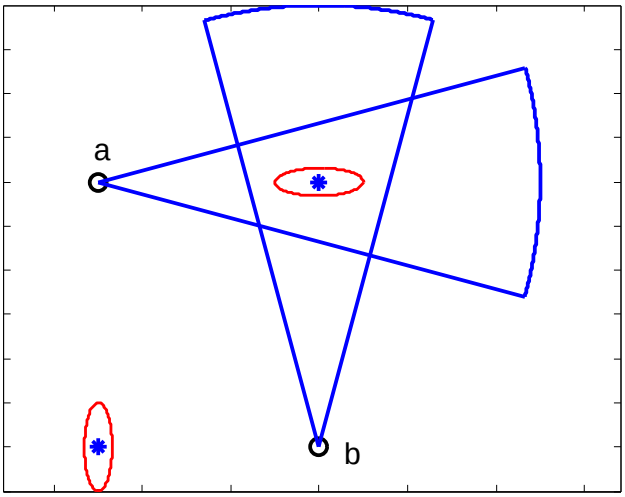

Fig. 2. Counterexample to show that our objective function is not continuous monotone (Case 1).

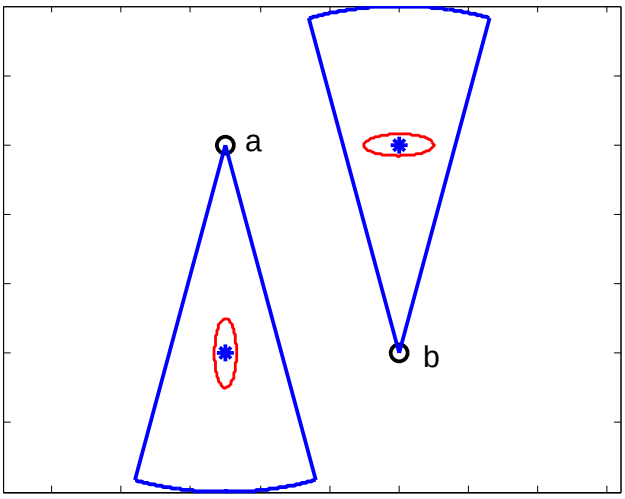

Fig. 3. Counterexample to show that our objective function is not continuous monotone (Case 2).

IV. Simulation Results and Further Discussion

We implement the heuristic approaches presented in the previous section in MATLAB for a scenario with six sensors and nine targets, where each sensor can take 10 possible directions $\{0, 2\pi/10, 2(2\pi/10), \ldots, 9(2\pi/10)\}$. Figures 4, 5, and 6 depict the locations of targets, sensors, and the solution from the approaches $\mathcal{H}_1$, $\mathcal{RH}_1$, and $\mathcal{MH}_1$ respectively. The solution from each of these approaches are the directions computed for the sensors (can be interpreted from the FOVs of sensors shown in these figures while using Figure 1 as reference). Table I compares the objective function values of the solutions obtained from each of the heuristic approaches discussed in this study along with the objective function value from continuous (relaxed) optimization. This table corroborates the result in Theorem 3.2 that the rollout on a heuristic approach outperforms the heuristic approach.

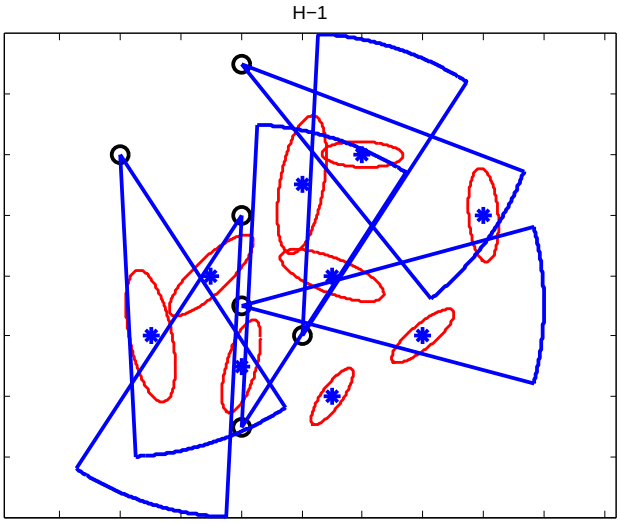

Fig. 4. Solution from $\mathcal{H}_1$

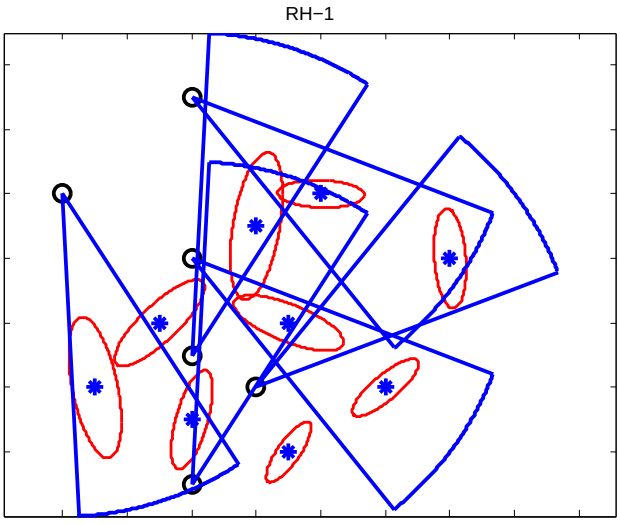

Fig. 5. Solution from $\mathcal{RH}_1$

Table I demonstrates that the objective function values from the heuristics are relatively close to that of the relaxed problem

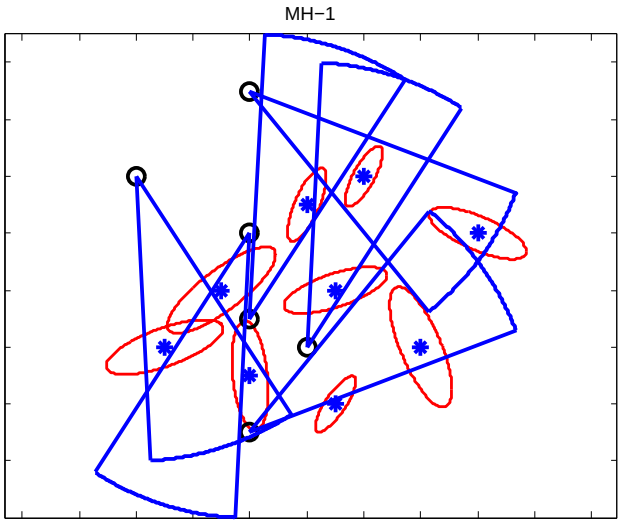

Fig. 6. Solution from $\mathcal{MH}_1$

TABLE I
OBJECTIVE FUNCTION VALUES FROM VARIOUS APPROACHES

Approach	**Objective function value**
Relaxed (upper bound)	46.67
$\mathcal{H}_1$	42.71
$\mathcal{RH}_1$	44.03
$\mathcal{H}_2$	43.76
$\mathcal{RH}_2$	44.93
$\mathcal{MH}_1$	41.06

(upper bound), which implies that the solutions from our heuristics are close to optimal. This prompts us to question if in fact we could have known beforehand that our heuristics are provably close to optimal because our problem exhibits properties that are known to result in provable suboptimality bounds. We will now investigate this question.

In a recent study [12], [13], it was shown that if an objective function is *string-submodular*[1], then the "greedy strategy" performs at least as good as $(1-1/e) \approx 0.63$ of the optimal. The definition of the above-mentioned greedy strategy in [12] is exactly the same as our heuristic algorithm $\mathcal{H}_1$. The following is an interesting observation from the results in our study. The objective function value from $\mathcal{H}_1$ is

$$R(u_{\mathcal{H}_1}) = 0.91 R_{\text{relaxed}} \geq 0.91 R_{\text{optimal}} \geq (1-1/e) R_{\text{optimal}},$$

i.e., the objective function value from algorithm $\mathcal{H}_1$ is at least as good as $(1-1/e)R_{\text{optimal}}$. We further compare the objective function values from $\mathcal{H}_1$ and the relaxed approach (which provides an upper bound on the optimal objective function value) for various scenarios with varying numbers of sensors and targets; Table II summarizes the results.

[1]An objective function is said to be string-submodular if it has the following properties: *forward-monotone* and *diminishing returns*. See [12] or [13] for the definition of these properties.

TABLE II
COMPARISON OF OBJECTIVE FUNCTION VALUES FROM $\mathcal{H}_1$ AND THE RELAXED APPROACH.

Scenario	$R(u_{\mathcal{H}_1})/R_{\text{relaxed}}$
3 sensors, 5 targets	0.92
3 sensors, 9 targets	0.90
4 sensors, 5 targets	0.77
4 sensors, 9 targets	0.92
6 sensors, 9 targets	0.91

Table II demonstrates that our heuristic algorithm $\mathcal{H}_1$ is at least as good as $(1-1/e)R_{\text{optimal}}$ for every scenario considered. This observation suggests that our objective function may have a string-submodular type property. Unfortunately, things are once again not quite as straightforward as one would have expected. Alas, our objective function turns out *not* to be string-submodular, as we show in the following counterexample. This suggests that our problem possesses highly nontrivial features that elude currently known analytical machinery.

Example Let us consider a scenario with three sensors and one target as shown in Figure 7. Let the sensors be a, b, and c. Without loss of generality, let $a \to b \to c$ be the sequence in which directions are computed. Let us assign $\pi/3, 3\pi/4, 3\pi/4$ to the sensors a, b, and c respectively. We can easily verify the following:

$$R((\pi/3,3\pi/4)) - R((\pi/3)) \leq R((\pi/3, 3\pi/4, 3\pi/4)) - R((\pi/3, 3\pi/4)),$$

where R is our objective function. The above inequality shows that our objective function does not have the diminishing returns property (again, see [12] or [13] for details), thus proving that our objective function is not string-submodular.

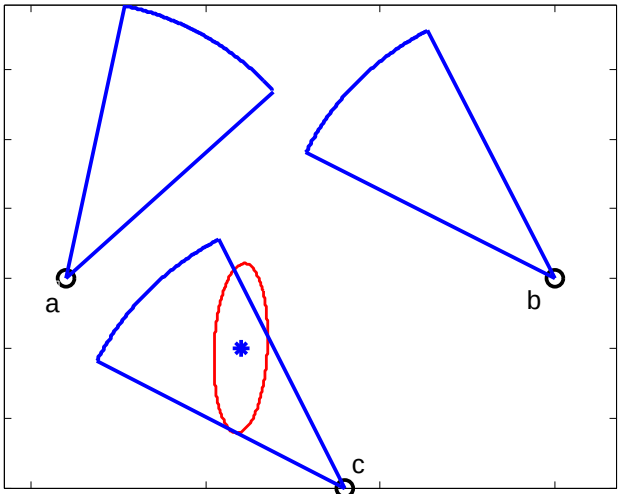

Fig. 7. Counterexample to show that our objective function is not string-submodular

V. Concluding Remarks

We investigated the problem of controlling the directional sensors for maximizing an information-gain-based objective function. We identified that this problem is a combinatorial optimization problem, and developed heuristic approaches ($\mathcal{H}_1$ and $\mathcal{H}_2$) to solve the problem approximately. We further improved the performance of our heuristics by applying an approximate dynamic programming approach called rollout. The rollout on our heuristic approach outperforms the heuristic approach, and our empirical results are in agreement with this.

We then addressed this problem via a different formulation, where the goal was to find an optimal mapping from the set of sensors to the set of targets that maximizes our information-gain-based objective function. This problem is also combinatorial in nature, so we extended the heuristic approach $\mathcal{H}_1$, developed for the previous formulation, to solve this mapping problem approximately, and we called this new heuristic algorithm $\mathcal{MH}_1$. We investigated natural sufficient conditions on objective functions that lead to provable guaranteed performance for our heuristics. Specifically, we proved that if an objective function is continuous monotone, then $\mathcal{MH}_1$ outperforms $\mathcal{H}_1$ with respect to the objective function value. However, we showed via a counterexample that our objective function does not satisfy this sufficient condition, suggesting that the nature of the problem is highly nontrivial. Nonetheless, the above result provides an analytical comparison of performances of $\mathcal{H}_1$ and $\mathcal{MH}_1$ for any generic continuous monotone objective function.

Our empirical results show that our heuristic algorithm $\mathcal{H}_1$ performed very close to the upper bound on the optimal, i.e., close to optimal, leading us to wonder whether our objective function possesses a property that would guarantee $\mathcal{H}_1$ to perform close to optimal. So, we investigated this question, and found a recent study in the literature that showed that if an objective function is string-submodular, then the greedy strategy ($\mathcal{H}_1$ in our study) performs at least as good as $(1 - 1/e)$ of the optimal. So, we further compared the performance of $\mathcal{H}_1$ for several scenarios with varying numbers of sensors and targets, and the results demonstrate that for each scenario the objective function value from $\mathcal{H}_1$ is at least as good as $(1 - 1/e)$ of the optimal. However, we found a counterexample proving our objective function is in fact not string-submodular. This again suggests that our problem possesses highly nontrivial features that elude currently known analytical machinery. It remains an interesting future study to understand why the performance of our heuristic algorithm is so close to optimal.

References

[1] S. Ragi, H. D. Mittelmann, and E. K. P. Chong, "Directional sensor control for maximizing information gain," in *Proc. SPIE 8857, Signal and Data Processing of Small Targets 2013*, San Diego, CA, Sep. 2013.

[2] Y. Wang and G. Cao, "Minimizing service delay in directional sensor networks," in *Proc. 2011 IEEE INFOCOM*, Shanghai, China, Apr. 2011, pp. 1790–1798.

[3] H. D. Ma and Y. H. Liu, "Some problems of directional sensor networks," *Int. J. Sensor Netorks*, vol. 2, pp. 44–52, 2007.

[4] J. Ai and A. A. Abouzeid, "Coverage by directional sensors in randomly deployed wireless sensor networks," *J. Comb. Optim.*, vol. 11, pp. 21–41, 2006.

[5] G. Fusco and H. Gupta, "Placement and orientation of rotating directional sensors," in *Proc. 7th Annu. IEEE Communications Society Conf. SECON*, Boston, MA, Jun. 2010.

[6] M. A. Guvensan and A. G. Yavuz, "On coverage issues in directional sensor networks: A survey," *Ad Hoc Networks*, vol. 9, pp. 1238–1255, 2011.

[7] D. P. Bertsekas, J. N. Tsitsiklis, and C. Wu, "Rollout algorithms for combinatorial optimization," *Journal of Heuristics*, vol. 3, pp. 245–262, 1997.

[8] R. Bellman, *Dynamic Programming*. Princeton, NJ: Princeton University Press, 1957.

[9] C. J. Costello and I. J. Wang, "Surveillance camera coordination through distributed scheduling," in *Proc. 44th IEEE Conf. Decision and Control*, Seville, Spain, Dec 2005, pp. 1485–1490.

[10] G. P. Kefalas, "A phased-array ground terminal for satellite communications," *IEEE Trans. Commun. Technol.*, vol. 13, no. 4, pp. 512–525, 1965.

[11] A. O. Hero, C. M. Kreucher, and D. Blatt, "Information theoretic approaches to sensor management," in *Foundations and Applications of Sensor Management*, A. O. Hero, D. Castanon, D. Cochran, and K. Kastella, Eds. New York: Springer, 2008.

[12] Z. Zhang, E. K. P. Chong, A. Pezeshki, W. Moran, and S. D. Howard, "Submodularity and optimality of fusion rules in balanced binary relay trees," in *Proc. 51st IEEE Conference on Decision and Control*, Maui, HI, Dec. 2012, pp. 3802–3807.

[13] Z. Zhang, Z. Wang, E. K. P. Chong, A. Pezeshki, and W. Moran, "Near optimality of greedy strategies for string submodular functions with forward and backward curvature constraints," in *Proc. 52nd IEEE Conference on Decision and Control*, Florence, Italy, Dec. 2013, pp. 5156–5161.